**ОГЛАВЛЕНИЕ**

[ВВЕДЕНИЕ 3](#_Toc37464681)

[1 ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ 4](#_Toc37464682)

[1.1 Анализ и выбор языка программирования 4](#_Toc37464683)

[1.2 Анализ и выбор среды разработки 7](#_Toc37464684)

[2 РАЗРАБОТКА ПРОГРАММЫ 10](#_Toc37464685)

[2.1 Вводимые и выводимые данные 10](#_Toc37464686)

[2.2 Системные требования 10](#_Toc37464687)

[2.3 Описание классов и методов 10](#_Toc37464688)

[3 ТЕСТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 15](#_Toc37464689)

[ЗАКЛЮЧЕНИЕ 17](#_Toc37464690)

[БИБЛИОГРАФИЧЕСКИЙ СПИСОК 18](#_Toc37464691)

[ПРИЛОЖЕНИЕ А. ЛИСТИНГ ПРОГРАММЫ 20](#_Toc37464692)

# **ВВЕДЕНИЕ**

игра

Тема данной курсовой работы – реализация программы для перевода числовой записи суммы в рублях в словесную. Во время создания программы будет использоваться подход ООП, поэтому важно формировать такую структуру программы, чтобы добавление новых компонентов при возможном дальнейшем ее развитии не влияли на уже существующие компоненты, или это влияние было минимальным.

Целью курсовой работы является разработка программы для перевода числовой записи числа в словесную на языке программирования высокого уровня.

Отсюда следуют задачи:

* анализ и выбор языка программирования;
* анализ и выбор среды разработки;
* получение практического опыта по разработке прикладных программ;
* овладение функционалом и инструментарием разработки на языке программирования высокого уровня.

# **1 ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ**

# **1.1 Анализ и выбор языка программирования**

Язык программирования – один из способов записи алгоритмов; совокупность набора символов системы, правил образования и истолкования конструкций из символов для задания алгоритмов с использованием символов естественного языка.

На текущий момент существует большое количество языков программирования. Ниже представлена малая их часть:

* Basic;
* Pascal;
* Delphi;
* Fortran;
* С и С++;
* Java;
* C#;
* Python;
* и т.д.;

Рассмотрим некоторые из них:

C++ — чрезвычайно мощный язык, содержащий средства создания эффективных программ практически любого назначения, от низкоуровневых утилит и драйверов до сложных программных комплексов самого различного назначения.

Достоинства:

Высокая совместимость с языком С, позволяющая использовать весь существующий С-код.

Поддерживаются различные стили и технологии программирования, включая традиционное директивное программирование, ООП, обобщенное программирование, метапрограммирование (шаблоны, макросы).

Имеется возможность работы на низком уровне с памятью, адресами, портами.

Возможность создания обобщённых контейнеров и алгоритмов для разных типов данных, их специализация и вычисления на этапе компиляции, используя шаблоны.

Кроссплатформенность. Доступны компиляторы для большого количества платформ, на языке C++ разрабатывают программы для самых различных платформ и систем.

Эффективность. Язык спроектирован так, чтобы дать программисту максимальный контроль над всеми аспектами структуры и порядка исполнения программы.

Недостатки:

Синтаксис, провоцирующий ошибки:

Препроцессор, унаследованный от С, очень примитивен.

Плохая поддержка модульности (по сути, в классическом Си модульность на уровне языка отсутствует, её обеспечение переложено на компоновщик). Подключение интерфейса внешнего модуля через препроцессорную вставку заголовочного файла (#include) серьёзно замедляет компиляцию при подключении большого количества модулей (потому что результирующий файл, который обрабатывается компилятором, оказывается очень велик).

C# нельзя рассматривать только как язык, это часть большой системы, которая включает в себя ОС Windows, среду разработки Visual Studio (и другие инструменты), ну и, конечно, саму Microsoft, которая обеспечивает поддержку этого языка. C# будет существовать и развиваться, пока существует Microsoft и пока люди пользуются Windows, а это будет ещё очень и очень долго.

Достоинства:

* легкость обучения;
* много синтаксического сахара. Синтаксический сахар - это такие конструкции, которые созданы для облегчения написания и понимания кода.
* возможность использовать его в написании как десктопных программ так и мобильных приложений, а так же сайтов.

Минусы:

* ориентированность, в основном, только на .NET (на Windows платформу);
* бесплатность только для небольших компаний и программистов одиночек.

Java-технологии имеют много особенностей, отличающие их от других технологий разработки программного обеспечения.

Достоинства:

* Переносимость. Программы, написанные на языке Java, после однократной трансляции в байт-код могут быть исполнены на любой платформе, для которой реализована виртуальная Java-машина. Наиболее эффективно возможности реального компьютера может использовать только программа, написанная с использованием «родного» машинного кода.
* Безопасность. Функционирование программы полностью определяется (и ограничивается) виртуальной Java-машиной. Отсутствуют указатели и другие механизмы для непосредственной работы с физической памятью и прочим аппаратным обеспечением компьютера. Дополнительные ограничения снижают возможность написания эффективно работающих Java-программ.
* Надежность. В языке Java отсутствуют механизмы, потенциально приводящие к ошибкам: арифметика указателей, неявное преобразование типов с потерей точности и т.п. Присутствует строгий контроль типов, обязательный контроль исключительных ситуаций. Многие логические ошибки обнаруживаются на этапе компиляции. Наличие дополнительных проверок снижает эффективность выполнения Java-программ.

Недостатки языка:

* Язык не полностью объектно-ориентирован: Язык не имеет таких свойств полностью объектно-ориентированного языка, как индивидуальные переменные, множественное наследование и.т.д.
* Соответствие корпоративным требованиям: Язык довольно популярен в мире программ с открытым кодом и технически превосходит многие коммерческие аналоги. Однако ему не хватает некоторых важных, с точки зрения корпоративной среды-особенностей. Это означает, что если мы захотим использовать PHP в корпорации, то это либо вообще не удастся сделать, либо потребуется значительно больше дополнительных программных средств, чем при использовании Java или C++.

Исходя из достоинств и недостатков языков программирования был выбран язык Java, так как он является самым безопасным из всех описанных языков и написание кроссплатформенного приложения на нем будет наиболее простым по сравнению с С++.

# **1.2 Анализ и выбор среды разработки**

Интегрированная среда разработки или по-английски Integrated development environment — IDE — это программа, которая содержит в себе инструменты для разработки программного обеспечения. Обычно современная среда разработки включает в себя:

* Текстовый редактор с подсветкой кода;
* компилятор или интерпретатор;
* браузер классов, инспектор объектов и диаграмму иерархии классов;
* средства автоматизации сборки;
* отладчик;
* средства для интеграции с системами управления версиями (Git);
* инструменты для упрощения конструирования графического интерфейса пользователя.

NetBeans – модульная среда разработки, и все функции IDE входят в пакеты, называемые модулями. Он предлагает легкую интеграцию с программным обеспечением для управления версиями. NetBeans был создан с открытым исходным кодом, поэтому каждый может внести свой вклад в его развитие. Он бесплатный и работает на большинстве операционных систем и платформ.

IntelliJ является закрытой исходной Java IDE и одной из главных среди IDE для Java. IntelliJ IDEA имеет отличный современный интерфейс. Он был выбран как базовая IDE для Android Studio – нового программного обеспечения для создания Android-приложений, разработанного компанией Google.

Существуют две версии IntelliJ – Community Edition, которая является бесплатной, и Ultimate Edition, которая полностью признана и требует использования оплачиваемых лицензий.

Eclipse – одна из самых популярных IDE, не только для Java, но и для C++ с PHP. Eclipse – это инструмент с открытым исходным кодом, имеющий отличное сообщество разработчиков. В нем также имеется огромная библиотека плагинов, созданная самими пользователями. У этой IDE есть множество версий, самая популярная – Eclipse Oxygen. Eclipse также была базовой IDE для старого Android SDK (Software Development Kit).

lueJ — это кроссплатформенная IDE, разработанная для образовательных целей и разработки небольших программ на Java. Она бесплатна, с открытым исходным кодом и предоставляет базовый набор инструментов разработки Java. Продукт имеет ряд интерактивных инструментов обучения и простой интерфейс по сравнению с большинством полноценных IDE. Основные функции включают выделение области видимости, проверку объектов и создание диаграмм исходного кода.

JDeveloper — полностью свободная среда разработки, выпущенная Oracle. JDeveloper предлагает несколько функций, таких как: рефакторинг; контроль версий Git; аудит и метрики; отладка и профилирование кода; поддержка Ant и Maven. Он поставляется в двух версиях: Studio Edition, который имеет полностью загруженный набор инструментов для создания программ на Java, и Java Edition, в котором отсутствуют функции J2EE, ADF, UML и базы данных, но эта версия занимает меньше места и быстрее работает. Java Edition имеет все основные функции, необходимые для Java, и должен удовлетворять большинству начинающих Java-программистов.

jCreator — это быстрая и эффективная Java IDE, написанная на C ++ (который предположительно делает его быстрее, чем другие IDE, написанные на Java). Он имеет интерфейс, похожий на более ранние версии Visual Studio, и позволяет настроить его по своему усмотрению. jCreator — платная IDE, но доступна бесплатная версия без таких функций, как автоматическое завершение кода. jCreator предлагает только поддержку Windows, но ее можно запустить на Linux с помощью Vine.

Изучив самые известные IDE для разработки на языке Java выбор пал на IDE NetBeans.

# **2 РАЗРАБОТКА ПРОГРАММЫ**

# **2.1 Вводимые и выводимые данные**

Одной из самых важных функций любого языка программирования является предоставление возможностей для управления программой вручную.

Выводимыми данными являются данные сообщаемые пользователю, а вводимыми являются те данные, которые пользователь сообщает программе.

Выводимые данные в программе представлены в виде диалоговых сообщений с просьбой о вводе числа и сообщении с результатом его обработки.

Вводимой информацией является ввод числа для последующего его преобразования в словесную форму.

# **2.2 Системные требования**

Требования к «железу»:

* 50 кб свободного пространства на жестком диске
* 40 мб оперативной памяти

Требования к операционной системе:

* JRE (виртуальная машина Java)

По вышесказанному описанию можно сделать вывод о том, что разработанная программа имеет понятный интерфейс с возможностью запуска на «слабых» компьютерах.

# **2.3 Описание классов и методов**

В процессе разработки программы били созданы два класса:

NumberInWords – класс, описывающий логику преобразования числовой формы записи в словесную.

Main – класс предназначенный для взаимодействия пользователя с логикой программы.

Поля класса NumberInWords:

– private BigDecimal amount – поле класса представляющие числовое значение числа.

– private boolean IsPositive – поле класса указывающие является ли число положительным.

Методы класса NumberInWords:

* Конструкторы:
  + public NumberInWords (long l)

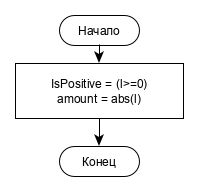


Рисунок 2.1 - Метод NumberInWords (long l)

* + public NumberInWords (double l)

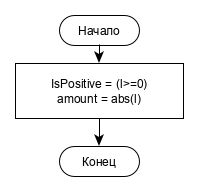


Рисунок 2.1 - Метод NumberInWords (double l)

* String morph(long n, String f1, String f2, String f5) – метод определяющий верное склонение слова.

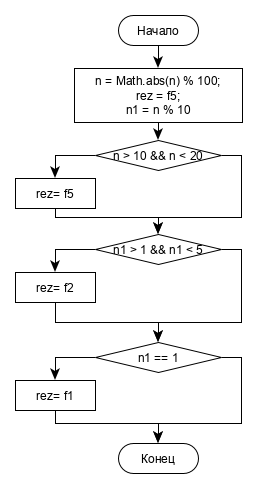


Рисунок 2.3 - Метод String morph(long n, String f1, String f2, String f5)

* public String toString() – переопределенный метод toString, реализующий алгоритм формирования словестной формы записи.

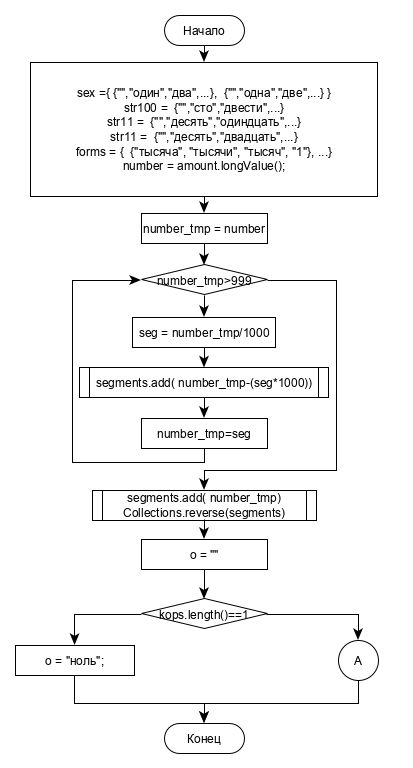


Рисунок 2.4 - Метод toString()

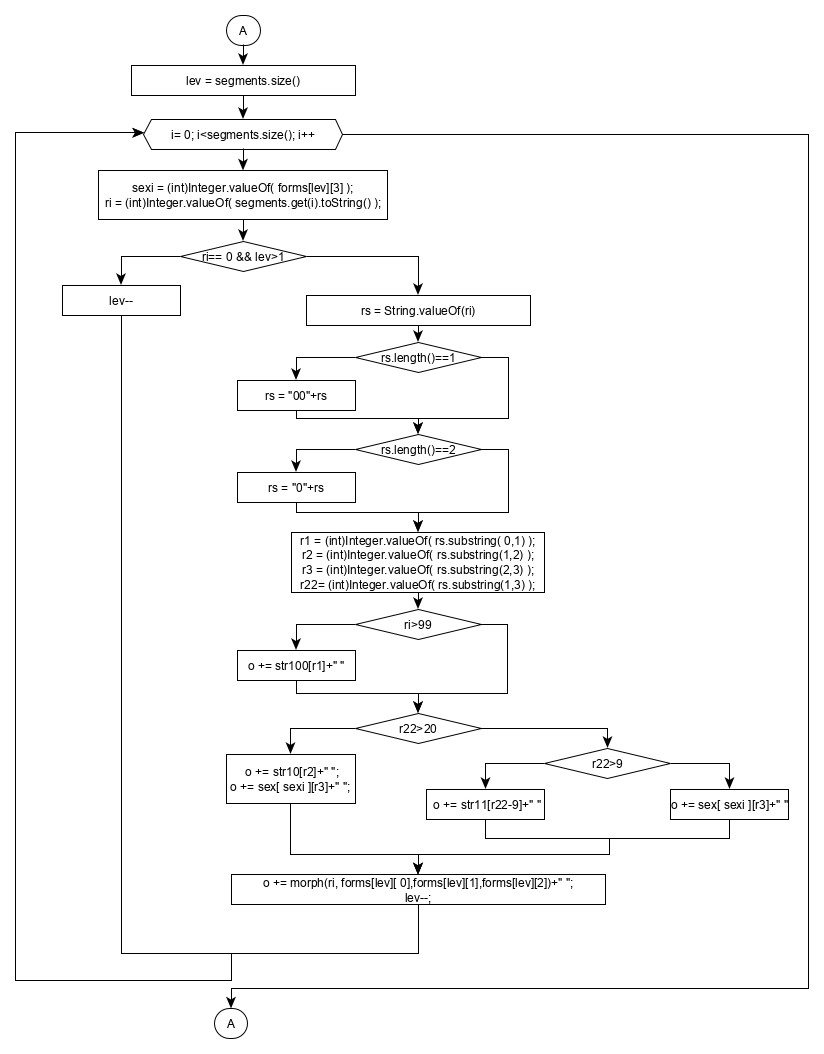


Рисунок 2.5 - Часть А алгоритма toString()

# **3 ТЕСТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ**

При разработке программ наиболее трудоемким является этап отладки и тестирования программ.

Цель тестирования, т.е. испытания программы, заключается в выявлении имеющихся в программе ошибок.

Цель отладки состоит в выявлении и устранении причин ошибок.

Разработанное приложение в процессе отладки проверялась на валидацию кода и на наличие ограничений, накладываемых на данные.

Выбор метода тестирование зависел от функциональных требований, предъявляемых к приложению и вариантов использования приложения.

Поэтому был выбран метод модульного тестирования.

Тестирование проводилось в ручном режиме путем последовательного ввода различных наборов данных.

Кроме того, проверялась работа приложения на разных устройствах и замечаний по его работе нет.

Отладка приложения проводилась по мере написания кода, тестирование выполнялось в ручном режиме.

При тестировании проводилась проверка:

* корректности обработки данных
* правильность отображения информации.

Проверка работоспособности программы, проведена на компьютерах с операционными системами семейства Windows и Linux с установленными виртуальными машинами java. Сбоев в работе программы не наблюдалось.

На рисунках 3.1 – 3.5 приведены примеры работы программы.



Рисунок 3.1 – Тест программы №1



Рисунок 3.2 – Тест программы №2



Рисунок 3.3 – Тест программы №3



Рисунок 3.4 – Тест программы №4



Рисунок 3.5 – Тест программы №5

# **ЗАКЛЮЧЕНИЕ**

В данной курсовой работе были выполнены все поставленные цели и задачи, а именно:

* получен практический опыт по разработке прикладных программ;
* был изучен функционал и инструментарий IDE NetBeans и языка программирования Java;
* реализована программа для перевода числовой формы записи в словесную с применением принципов ООП.

Так же было произведено тестирование программы, по результатом тестирования можно судить о отсутствии ошибок в работе программы.

# **БИБЛИОГРАФИЧЕСКИЙ СПИСОК**

1. Альфред, В. Ахо Компиляторы. Принципы, технологии и инструментарий / Альфред В. Ахо и др. - М.: Вильямс, 2015. - 689 c.
2. Берд, Барри Java для чайников / Барри Берд. - М.: Диалектика / Вильямс, 2013. - 521 c.
3. Гарнаев, Андрей WEB-программирование на Java и JavaScript / Андрей Гарнаев , Сергей Гарнаев. - Москва: СПб. [и др.] : Питер, 2017. - 718 c.
4. Гонсалвес, Энтони Изучаем Java EE 7 / Энтони Гонсалвес. - М.: Питер, 2016. - 640 c.
5. ГОСТ 19.401 – 78. ЕСПД. Текст программы. Требования к содержанию и оформлению. - М.: Изд-во стандартов, 1978.
6. ГОСТ 19.505 – 79. ЕСПД. Руководство оператора. Требования к содержанию и оформлению. - М.: Изд-во стандартов, 1977.
7. ГОСТ 19.701 - 90. ЕСПД. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения. - М.: Изд-во стандартов. - 1990.
8. Гупта, Арун Java EE 7. Основы / Арун Гупта. - М.: Вильямс, 2014. - 336 c.
9. Монахов, В. Язык программирования Java и среда NetBeans (+ CD-ROM) / В. Монахов. - М.: БХВ-Петербург, 2012. - 720 c.
10. Савитч, Уолтер Язык Java. Курс программирования / Уолтер Савитч. - М.: Вильямс, 2015. - 928 c.
11. Хабибуллин, Ильдар Самоучитель Java / Ильдар Хабибуллин. - М.: БХВ-Петербург, 2014. - 768 c.
12. Шилдт, Герберт Java 8. Руководство для начинающих / Герберт Шилдт. - М.: Вильямс, 2015. - 720 c.
13. Эккель, Брюс Философия Java / Брюс Эккель. - М.: Питер, 2016. - 809 c.

# **ПРИЛОЖЕНИЕ А. ЛИСТИНГ ПРОГРАММЫ**

// NumberInWords.java

package numberinwords;

import java.util.ArrayList;

import java.util.Collections;

import java.math.BigDecimal;

public class NumberInWords {

private BigDecimal amount;

private boolean IsPositive;

public NumberInWords(long l) {

this.IsPositive = (l>=0);

this.amount = new BigDecimal( Math.abs(l));

}

public NumberInWords(double l) {

this.IsPositive = (l>=0);

this.amount = new BigDecimal( Math.abs(l));

}

public String toString() {

String[][] sex = {

{"","один","два","три","четыре","пять","шесть","семь","восемь","девять"},

{"","одна","две","три","четыре","пять","шесть","семь","восемь","девять"},

};

String[] str100= {"","сто","двести","триста","четыреста","пятьсот","шестьсот","семьсот", "восемьсот","девятьсот"};

String[] str11 = {"","десять","одиннадцать","двенадцать","тринадцать","четырнадцать", "пятнадцать","шестнадцать","семнадцать","восемнадцать","девятнадцать","двадцать"};

String[] str10 = {"","десять","двадцать","тридцать","сорок","пятьдесят","шестьдесят", "семьдесят","восемьдесят","девяносто"};

String[][] forms = {

{"тысяча", "тысячи", "тысяч", "1"},

{"миллион", "миллиона", "миллионов", "0"},

{"миллиард","миллиарда","миллиардов","0"},

{"триллион","триллиона","триллионов","0"},

//сюда можно добавить разряды

};

long number = amount.longValue();

long number\_tmp = number;

ArrayList segments = new ArrayList();

while(number\_tmp>999) {

long seg = number\_tmp/1000;

segments.add( number\_tmp-(seg\*1000) );

number\_tmp=seg;

}

segments.add( number\_tmp );

Collections.reverse(segments);

String o = "";

if (number== 0)

{

o = "ноль";

}

else

{

int lev = segments.size();

for (int i= 0; i<segments.size(); i++ )

{

int sexi = (int)Integer.valueOf( forms[lev][3] );

int ri = (int)Integer.valueOf( segments.get(i).toString() );

if (ri== 0 && lev>1)

lev--;

else

{

String rs = String.valueOf(ri);

if (rs.length()==1) rs = "00"+rs;

if (rs.length()==2) rs = "0"+rs;

int r1 = (int)Integer.valueOf( rs.substring( 0,1) );

int r2 = (int)Integer.valueOf( rs.substring(1,2) );

int r3 = (int)Integer.valueOf( rs.substring(2,3) );

int r22= (int)Integer.valueOf( rs.substring(1,3) );

if (ri>99) o += str100[r1]+" ";

if (r22>20)

{

o += str10[r2]+" ";

o += sex[ sexi ][r3]+" ";

}

else {

if (r22>9)

o += str11[r22-9]+" ";

else

o += sex[ sexi ][r3]+" ";

}

if(lev>1)

o += morph(ri, forms[lev-2][ 0],forms[lev-2][1],forms[lev-2][2])+" ";

lev--;

}

}

}

if (!IsPositive)

return "минус "+o;

else

return o;

}

private static String morph(long n, String f1, String f2, String f5) {

n = Math.abs(n) % 100;

String rez = f5;

long n1 = n % 10;

if (n > 10 && n < 20) rez= f5;

if (n1 > 1 && n1 < 5) rez= f2;

if (n1 == 1) rez= f1;

return rez;

}

}

//Main.java

package numberinwords;

import java.util.Scanner;

public class Main

{

public static void main(String[] args)

{

Scanner in = new Scanner(System.in);

System.out.print("Введите число: ");

int amount = in.nextInt();

NumberInWords miw = new NumberInWords(amount);

System.out.println("Стрококое представление числа: " + miw);

}

}